PROJECTE MONGODB

Bases de Dades no Relacionals

Cèlia Martínez
Marta Matute
Goretti Pena
Judit Ugas

Durant aquest projecte hem treballat el disseny, la implementació i la consulta d'una base de dades proporcionada en format Excel. L'esquelet d'aquest informe ve donat per una primera part on expliquem el disseny en el que ens hem basat a l'hora de fer la implementació i les col·leccions que hem definit (exercici 1), una segona part on s'explica breument el codi de l'script de python que llegeix les dades i crea una base de dades a MongoDB (exercici 2), i finalment un joc de proves amb 9 consultes amb les solucions obtingudes (exercici 3). Al final hi ha afegida també una breu descripció del repartiment de la feina entre les membres de l'equip. Tota la documentació pertinent al projecto es pot trobar al reposotori de GitHub Projecte-MongoDB-Grup2.

Exercici 1. Disseny.- El disseny que hem pensat per a convertir el model Entitat-Relació a un conjunt de col·leccions, hem considerat 4 col·leccions: "Method", "Patient", "Nodules" i "CtScanner".

En la col·lecció "Mehod" hem guardat la informació corresponent a l'entitat "Method" del disseny Entitat-Relació sobre el que havíem de basar. Aquesta col·lecció està formada pels atributs "MethodID" (clau primària), "FeatSelection", "FeatDescription", "Classifier" i "Experiment". Aquest últim atribut a la vegada representa la entitat del diagrama amb el mateix nom. Per això conté una llista amb els següents atributs: "Repetition", "Train, "BenignPrec", "BenignRec", "MalignPrec" i "MalignRec". En llegir el document de dades, si trobem un id de "Method" que ja havíem llegit amb anterioritat, aleshores afegeix el nou experiment a la llista d'experiments.

En la col·lecció "Patient" vam guardar la informació de l'entitat "Patient" del diagrama. Els atributs que guardem per aquesta col·lecció son "PatientID" (clau primària), "Age", "Gender", "DiagnosisPatient" i "Nodules". Aquesta última torna a ser una representació de una de les entitats del disseny. Per aquest motiu conté una llista d'atributs, que son: "NoduleID" i "DiagnosisNodul". D'aquesta manera, cada cop que ens topem amb un pacient que ja havíem registrat dins la nostre base de dades, podrem afegir un nou nòdul afegint-lo a la llista de "Nodules".

Dins la col·lecció "Nodules", hem guardat la resta de la informació pertinent a l'entitat "Nodule" (i.e. "NoduleID" (clau primària), "DiagnosisNodul", i "Position" en forma de llista on hi guardem les posicions X, Y i Z) a més de relacionar-la amb l'entitat "CTScanner" (fent servir l'atribut "CTScanner", que en forma de llista conté els elements "CTID" i "Diameter"), l'entitat "Patient" (fent servir la segona clau primària "PatientID") i l'entitat (però no col·lecció) "Experiment". En aquest cas apliquem el patró subconjunt ja que ens permet accedir de forma més facil a la informació que considerem més rellevant i que consultarem més freqüentment.

Finalment, la col·lecció "CTScanner" està formada pels atributs "CTID" (clau primària), "Device", "dataCT", "ResolutionT", "ResolutionTV" i "ResolutionTC".

D'aquesta manera quedan relacionades totes les entitats del disseny tal com volíem.

Exercici 2. Script de python.- Per a realitzar la lectura en si de la base de dades desde un arxiu `.xlsx` i exportar-la a MongoDB hem creat un script de python que llegeix les dades de manera automàtica. Donat que no hem afegit cap mena d'argument complementari a l'script, per a executar el codi només cal fer servir la instrucció corresponent per a cada operador del sistema. Aquesta instrucció seria

```
$ python3 data_reading.py
```

per a Windows, i

```
$ data_reading.py
```

per a Linux.

L'scirpt en sí consta de 3 parts: la connexió al local host, la inicialització de les col·leccions, i la lectura del fitxer de dades.

Connexió al localhost: fem servir la llibreria pymongo per a connectar amb el localhost del nostre ordinador i importar les col·leccions a la base de dades que crearem fent servir noSQL. El codi, molt breu, queda de la forma:

```
# En execució remota
Host = 'localhost'
Port = 27017

DSN = "mongodb://{}:{}".format(Host,Port)

conn = MongoClient(DSN)

bd = conn['Cancer']
```

Inicialització de les col·leccions: per a cadascuna de les quatre col·leccions que hem creat, fem servir un if-else statement de manera que si la col·lecció no existeix dins la nostra base de dades, aleshores la creem, i en cas que existeixi, la eliminem. Aquesta última part es necessària per assegurar-nos que si el programa s'executa més d'una vegada, o s'interromp a meitat i es torna a executar, que no s'insereixen els valors per duplicat. Es mostra a continuació l'execució d'aquest codi per a una sola de les entitats (la resta de les entitats es fa de la mateixa manera canviant el nom):

```
if "Method" not in bd.list_collection_names():
    methods = bd.create_collection('Method')
else:
    methods = bd["Method"]
    methods.drop()
```

Lectura del fitxer de dades: Per a cadascuna de les col·leccions hem fet servir un procés molt similar. Vegem pas a pas aquest procés per a una de les entitats:

El primer de tot ha estat per a tots els casos la definició del diccionari on guardarem tota la informació.

```
Method = {}
```

A continuació obrim l'arxiu per a llegir-lo i iterem per les files del nostre arxiu de dades i per a cadascuna de les files definim una clau primària. Aquesta clau primària ha correspost en cadascun dels casos a les columnes de l'arxiu de dades "MethodID", "PatientID" i "CTID" per a les col·leccions Method, Patient i Scanner respectivament, i una clau primària composta per "NodulID" i "PatientID" en el cas de la col·lecció Nodules. Veiem a continuació el codi per a una de les col·leccions. La resta són anàlogs:

```
with open('Dades.xlsx', mode='rb') as fname:
    dfe = pd.read_excel(fname, sheet_name='MethodOutput')

for rowid in range(len(dfe)):
    row = dfe.iloc[rowid]

    my_id = row["MethodID"]
```

Seguidament, i encara dins el bucle, comprovem si la clau primària que volem afegir es troba ja dins del diccionari o no, i depenent del cas, creem una clau al diccionari, o bé en modifiquem una afegint nova informació a una instància ja existent. L'estructura general d'aquesta part en codi ha sigut:

```
if my_id not in Method:
        Method[my_id] = {…} # Aquí afegiriem tots els atributs que calgués

    else:
        new_dict = {…}
        Method[my_id]['Experiment'].append(new_dict)
```

Finalment afegim cadascun dels valors del diccionari que hem creat dins la base de dades i tanquem la connexió fent servir les comandes:

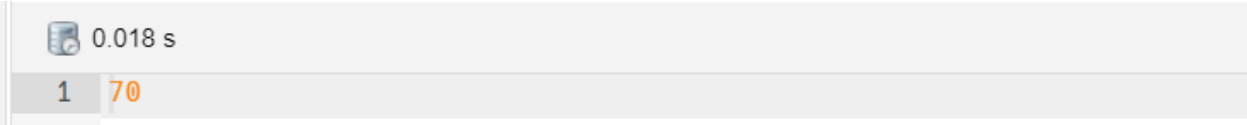
```
for id_dict in Method.keys():
    methods.insert_one(Method[id_dict])
# També el mateix per a la resta de col·leccions

conn.close()
```

Exercici 3. Consultes.-

```
/* 1. Escàners diferents que hi ha a la BD. Mostra el device.*/
db.CtScanner.distinct("Device")
```

Key	Value	Type
(1)	Fujifilm Holdings	String
(2)	GE Healthcare	String
(3)	Siemens Sensation	String
(4)	Toshiba Aquilon One	String

```
/*2. Número total de nòduls que s'han utilitzat per l'entrenament
     (train=1) de l'experiment 1 del mètode "Method2".*/
db.Nodules.find({ "Experiment": { $elemMatch: { "Train": 1, "MethodID": "Method2" ,
"ExperimentRepetition": 1} } }).count()
```




```
/*3. Valor màxim, mínim i mitjà de BenignPrec agrupat per classificador (classifier).
     Mostra ID del mètode, MaxBenignPrec, MinBenignPrec, AvgBenignPrec.*/
db.Method.aggregate([
    {$unwind: "$Experiment"},
    {$group: { _id: "$Classifier", MaxBenignPrec:{$max:"$Experiment.BenignPrec"},
MinBenignPrec:{$min:"$Experiment.BenignPrec"}, AvgBenignPrec:{$avg:"$Experiment.BenignPrec"}}}
    ])
```

Key	Value	Type
(1) Logit	{ MaxBenignPrec : 39.4736842105263, MinBenignPrec : 0, AvgBenignPrec : 31.521027350987403 } (4 fields)	Document
_id	Logit	String
MaxBenignPrec	39.4737	Double
MinBenignPrec	0	Double
AvgBenignPrec	31.521	Double
(2) SVM	{ MaxBenignPrec : 66.6666666666667, MinBenignPrec : 27.5, AvgBenignPrec : 38.25809070621295 } (4 fields)	Document
(3) NN	{ MaxBenignPrec : 44.4444444444444, MinBenignPrec : 30.188679245283, AvgBenignPrec : 35.599680594762695 }	Document

```
/*4. Numero total d'homes i dones. Mostra sexe i número total. */
db.Patient.aggregate([
    {$group: { _id: "$Gender", count:{$sum:1}}}])
```

Key	Value	Type
(1) Man	{ count : 63 }	Document
_id	Man	String
count	63	Int32
(2) Woman	{ count : 38 }	Document

```
/*5. Pacients amb més de dos nòduls. Mostra ID del Pacient, sexe, edat, diagnòstic del Pacient*/
db.Patient.aggregate([
    {$unwind:"$Nodules"},
    {$match:{'Nodules.NoduleID':3}},
    {$project: {
                _id:0,
                PatientID: 1,
                Age: 1,
                Gender: 1,
                DiagnosisPatient: 1 }
            }
])
```

Key	Value	Type
(1)	{ PatientID : "LIDC-IDRI-0174", Age : 75, Gender : "Man", DiagnosisPatient : "Malign" } (4 fields)	Object
PatientID	LIDC-IDRI-0174	String
Age	75	Int32
Gender	Man	String
DiagnosisPatient	Malign	String
(2)	{ PatientID : "LIDC-IDRI-0194", Age : 60, Gender : "Man", DiagnosisPatient : "Malign" } (4 fields)	Object
(3)	{ PatientID : "LIDC-IDRI-1011", Age : 69, Gender : "Woman", DiagnosisPatient : "Malign" } (4 fields)	Object

```
/*6. Mostrar els 4 mètodes amb més repeticions de l'experiment.
     Mostra el ID del Mètode i número de repeticions de l'experiment.*/
db.Nodules.aggregate([
    {$unwind: "$Experiment"},
    {$group: {_id: "$Experiment.MethodID", maxQuantity: { $max: "$Experiment.ExperimentRepetition"
} }},
    {$project: {"MethodID": 1, maxQuantity: 1}},
    {$sort: {maxQuantity: -1}},
    {$limit: 4}
    ])
```

Key	Value	Type
(1) Method2	{ maxQuantity : 5 }	Document
_id	Method2	String
maxQuantity	5	Int32
(2) Method9	{ maxQuantity : 5 }	Document
(3) Method13	{ maxQuantity : 5 }	Document
(4) Method16	{ maxQuantity : 5 }	Document

```
/*7. Per cada pacient els escàners (CTs) que s'ha fet.
     Mostra el ID del Pacient, device i la data del CT.*/
db.Nodules.aggregate([
    {$lookup:
        {
            from:"CtScanner",
            localField:"CtScanner.CTID",
            foreignField:"CTID",
            as:"Scanner"
        }
    },
    {$unwind:"$Scanner"},
    {$project:{PatientID:1, "Scanner.Device":1, "Scanner.dataCT":1, _id:0}}
    ])
```

Key	Value	Type
(1)	{ PatientID : "LIDC-IDRI-0071" } (2 fields)	Object
PatientID	LIDC-IDRI-0071	String
Scanner	{ Device : "Siemens Sensation", dataCT : ISODate("2018-07-15T02:00:00.000+02:00") }	Object
(2)	{ PatientID : "LIDC-IDRI-0072" } (2 fields)	Object
(3)	{ PatientID : "LIDC-IDRI-0090" } (2 fields)	Object
(4)	{ PatientID : "LIDC-IDRI-0091" } (2 fields)	Object
(5)	{ PatientID : "LIDC-IDRI-0100" } (2 fields)	Object
(6)	{ PatientID : "LIDC-IDRI-0124" } (2 fields)	Object
(7)	{ PatientID : "LIDC-IDRI-0129" } (2 fields)	Object
(8)	{ PatientID : "LIDC-IDRI-0135" } (2 fields)	Object
(9)	{ PatientID : "LIDC-IDRI-0137" } (2 fields)	Object

Nombre total de files: 117

```
/*8. Mostrar els pacients que tenen tots els seus nóduls amb diagnosis = "Benign"
     i el seu recompte.*/
db.Patient.aggregate([
    {$match:{DiagnosisPatient:'Benign'}},
    {$unwind:"$Nodules"},
    {$group:{ _id:"$PatientID", NumNodules:{$max: "$Nodules.NoduleID"}}}
    ])
```

Key	Value	Type
(1) LIDC-IDRI-0071	{ NumNodules : 1 }	Document
_id	LIDC-IDRI-0071	String
NumNodules	1	Int32
(2) LIDC-IDRI-0247	{ NumNodules : 1 }	Document
(3) LIDC-IDRI-0286	{ NumNodules : 1 }	Document
(4) LIDC-IDRI-0180	{ NumNodules : 1 }	Document
(5) LIDC-IDRI-0256	{ NumNodules : 1 }	Document
(6) LIDC-IDRI-0277	{ NumNodules : 1 }	Document
(7) LIDC-IDRI-0289	{ NumNodules : 2 }	Document
(8) LIDC-IDRI-0100	{ NumNodules : 1 }	Document
(9) LIDC-IDRI-0279	{ NumNodules : 1 }	Document
(10) LIDC-IDRI-0072	{ NumNodules : 1 }	Document
(11) LIDC-IDRI-0257	{ NumNodules : 1 }	Document
(12) LIDC-IDRI-0283	{ NumNodules : 1 }	Document
(13) LIDC-IDRI-0171	{ NumNodules : 1 }	Document
(14) LIDC-IDRI-0205	{ NumNodules : 1 }	Document
(15) LIDC-IDRI-0273	{ NumNodules : 1 }	Document
(16) LIDC-IDRI-0178	{ NumNodules : 1 }	Document
(17) LIDC-IDRI-0185	{ NumNodules : 1 }	Document

Nombre total de files: 35

```
/*9. Modificar la ResolutionTV aumentant-la un 20% dels escàners que es van realitzar
     amb DataCT = 18/11/2018*/
db.CtScanner.aggregate([
    {$match:{dataCT:ISODate("2018-07-15T00:00:00.0Z")}},
    {$project :{CTID:1, Device:1, dataST:1, ResolutionT:1, ResolutionTC:1, half_price :{ $multiply
:[1.2, "$ResolutionTV"]}}}
    ])
```

Key	Value	Type
(1) 624c10e212e638d303a346f9	{6 fields}	Document
_id	624c10e212e638d303a346f9	ObjectId
CTID	1	Int32
Device	Siemens Sensation	String
ResolutionT	0.625	Double
ResolutionTC	206	Int32
half_price	144	Double
(2) 624c10e212e638d303a3472a	{6 fields}	Document

Repartició de Tasques:

La creació del disseny va ser tasca conjunta entre la Cèlia, la Judit i la Goretti. La implementació del codi va ser feina conjunta de tot l'equip mentre fèiem una trucada (Goretti, Judit, Cèlia i Marta). Les quèries va ser feina majoritàriament de la Cèlia amb ajuda de la Judit i la Goretti, i de l'informe i la creació del Github se'n va ocupar la Marta.